**Formative Assessment: NLP - Emotion Classification in Text**

**Due on 11th October 24**

**Submitted by Aiswarya Jayaprakash**

Objective:   
Develop machine learning models to classify emotions in text samples.  
  
Dataset:  
<https://drive.google.com/file/d/1HWczIICsMpaL8EJyu48ZvRFcXx3_pcnb/view?usp=drive_link>  
  
Key components to be fulfilled :  
1. Loading and Preprocessing (3 marks)

* Load the dataset and perform necessary preprocessing steps. This should include text cleaning, tokenization, and removal of stopwords. Explain the preprocessing techniques used and their impact on model performance.



**Preprocessing**





Preprocessing includes steps like text cleaning, tokenization, and removal of stopwords.

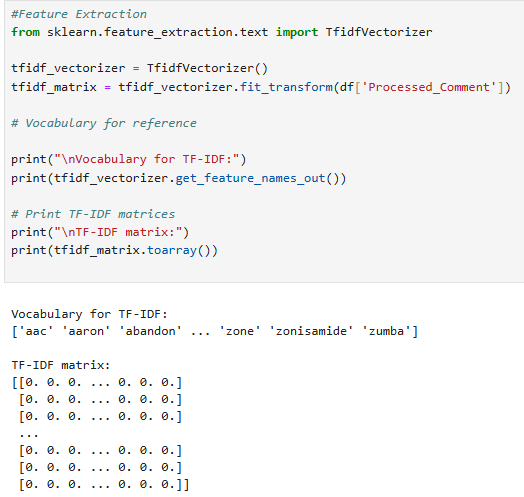
Text cleaning process converts all characters to lowercase and standardises the text, reducing the number of unique words. This improves efficiency by ensuring different cases of the same word are treated equally. Removal of Short Words removes words that are one or two characters long as short words often carry less meaningful information. Hence the data noise get reduced focusing on more informative words, enhancing model performance. Further, removal of punctuation usually doesn't carry significant meaning in text analysis and simplifies text, leading to better feature extraction.

Tokenization and Stopword Removal

Tokenization splits text into individual words (tokens) and removes common stopwords (e.g., "the", "is"). Tokenization breaks down text into manageable pieces, while stopwords often carry little semantic information. This reduces dimensionality and noise, focusing on more informative words. Hence, ceaned text improves feature extraction and model accuracy by removing noise and reducing the vocabulary size.

2. Feature Extraction (2 marks):

* Implement feature extraction using CountVectorizer or TfidfVectorizer. Describe how the chosen method transforms the text data into numerical features.

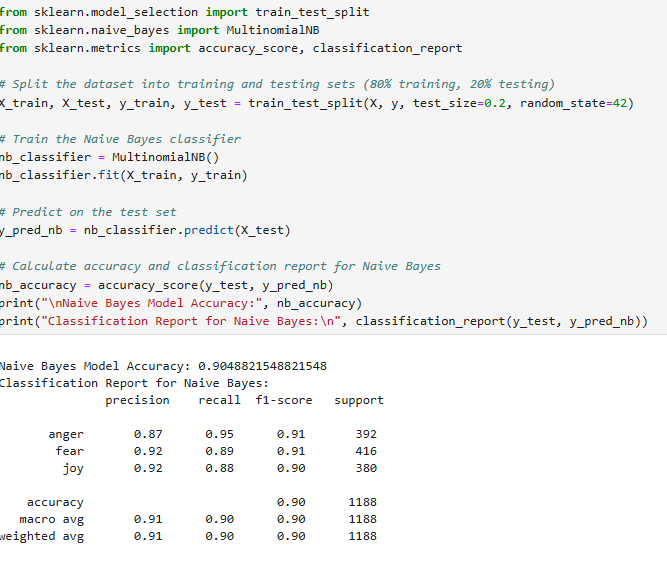
 TfidfVectorizer calculates a matrix where each word is weighted based on its term frequency (how often it appears in a document) and inverse document frequency (how common it is across all documents). It works by counting the occurrences of each word first (like CountVectorizer) but then applies a weighting formula. The formula decreases the weight of common words and increases the weight of rare words, providing a measure of word importance in each document.

3. Model Development (2 marks):

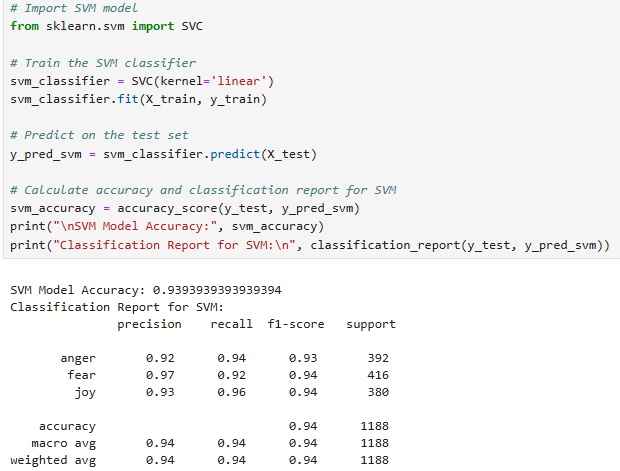
* Train the following machine learning models

**a)Naive Bayes**

Naive Bayes probabilistic model works well for text classification by assuming independence between features (words). It’s fast and efficient for text data.

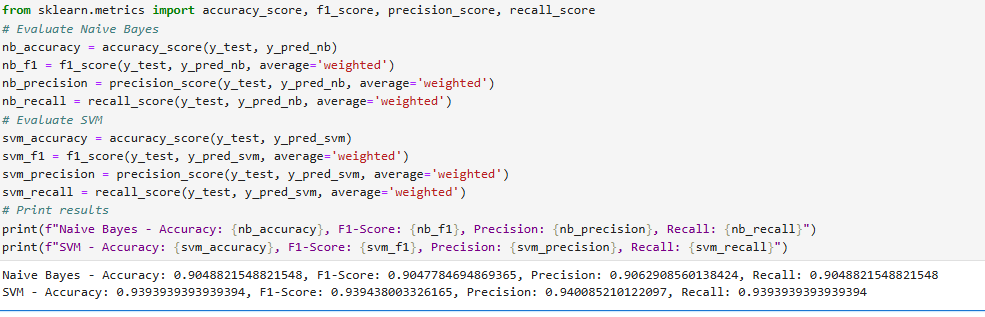


**b)Support Vector Machine**  
SVM works by finding the hyperplane that maximally separates the data into different classes. It is useful for high-dimensional text data and can handle large feature spaces like TF-IDF.

****

4. Model Comparison (2 marks)

* Evaluate the model using appropriate metrics (e.g., accuracy, F1-score). Provide a brief explanation of the chosen model and its suitability for emotion classification.



Naive Bayes is fast and provides a good baseline but might oversimplify text features. SVM generally achieves higher performance in emotion classification by leveraging its ability to model complex boundaries bet